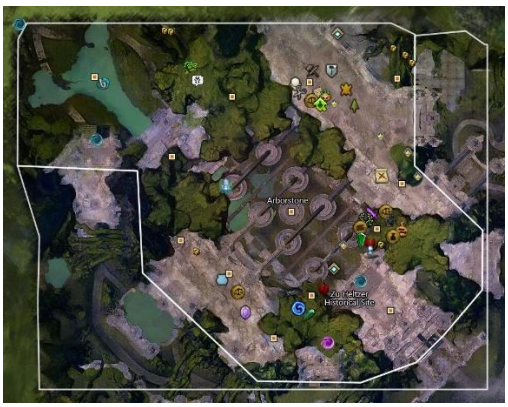
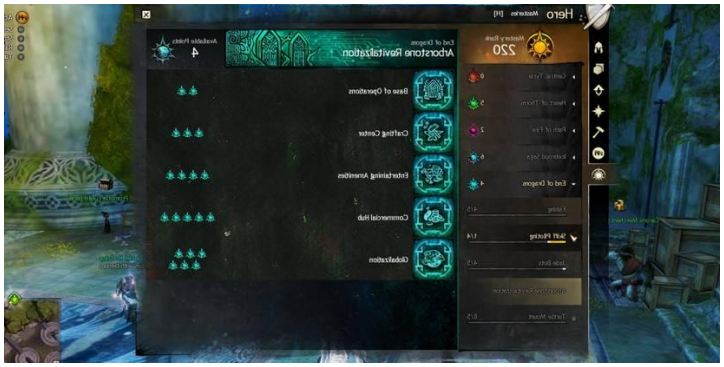
Annie VanderMeer – Sandbox Design

EXAMPLE #1 – *Guild Wars 2: End of Dragons*, Arborstone

Arborstone is a player hub that was introduced in the *End of Dragons* expansion. It was designed not only to evolve over the course of the expansion's story, but also expand in functionality as the player put points into a related skill tree. I coordinated with multiple departments to achieve this: speaking with the Systems/Progression team to make certain the unlocked functions and services were not only satisfying but well located within the area; collaborating with the Narrative team to place scenes and make sure that spawns and story moments triggered as needed; setting up spaces and spawns for the Story team to create a special area; and working with the world content team to add reactive spots and moments to follow NPC stories and collection progressions.

In addition to this, I placed the vast majority of the NPCs and set up their movements and schedules, filled out placeholder dialogue for all spawns, set up all functions (both default and progression-related), worked out kinks with movement through the space (a LOT of teleporters), and even developed and placed a player activity or two – including working with the Systems team to develop a satisfying Boss-style encounter. I also coordinated with other designers interested in the space to include content they wanted to develop – a jumping puzzle and a PvP area – and helped test and support these additions to the space.





EXAMPLE #2 – *Astroneer*, Procedural World Design and Supporting Systems

Being that all of the areas in *Astroneer* were procedurally generated, this presented a unique challenge to be tackled by almost all of the systems in the game. This wasn't just affected by what the player found on the surface level of the planets, but by *depth*, since digging is a core feature of the game, and reaching the cores of all seven planets is the only way to complete it. I was put in charge of managing the settings for the procedural elements, which were controlled by a proprietary system within Unreal 4 – and these settings didn't just affect standard design elements like hazards, puzzles, and items, but even foliage, depth and frequency of chasms, and so on.

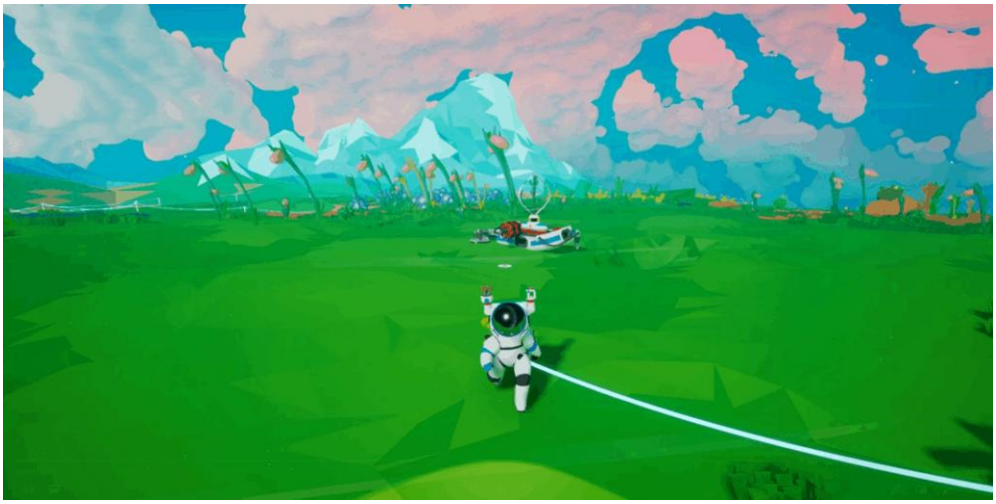
To make sense of what – in-engine – was an array of strange elements and associated values, I spent time establishing organizational categories. This covered things like: layers of planet depth, from surface to core; biome types on the surface (3-4 per planet); and scales of sun and wind (as ways the player gained energy). I documented these categories in Google Sheets with both the idealized values (ex. Low, Medium, High) and the raw in-engine values alongside them, so anyone who wanted to get a sense of how anything was distributed could scan a single document instead of trying to figure out an arcane system within UE4.

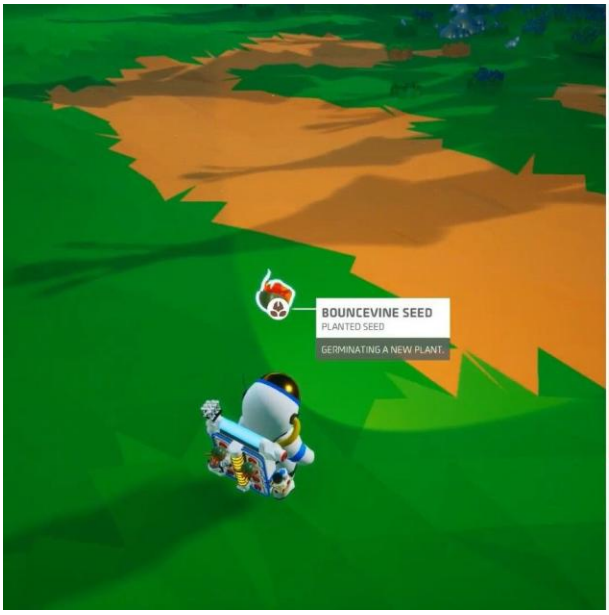
In addition to handling all the core values of procedural generation, I also handled several supplemental systems that impacted progression.

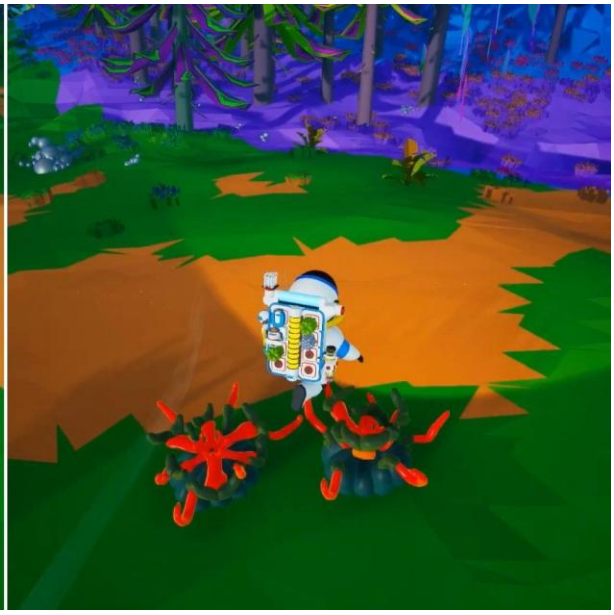
- **Hazards** were flora that were not only scaled by planet type, but also depth and biome, and I named them; worked with Programming to set up their basic AI procedures, scale things like damage and range; and set up the loot scales of potential rewards when they were dug up. I also expanded the initial Seed system to include "Mutant" (harmless) versions of Offensive Flora – allowing players to plant them for decoration – and the Spookysquash, a limited seasonal variant.
- **Discoveries** not only served a narrative purpose (implying the Astroneers before the player who had somehow disappeared or met with misfortune) but had to scale in terms of type in order to maintain progression. To do this, I created a layered system of blueprints that enabled me to start with a base item (say, a broken rover) and create a child item that had different tiers of loot that spawned on it, so that it was scaled correctly in reward to the difficulty level of the planet it was on. I also used layers of these items to build out "sets" – such as a Rover Station, a large scale landing pad, etc. – and themed them appropriate to each planet. This system also linked into another one I launched, Scrap, which allowed players to not only recycle their own items, but also Discoveries – including ones that persisted from very early versions of the game – in order to give them updated worth and utility and not just make them set dressing (because functionally speaking, *nothing* in that game is just set dressing!)
- **Harvestables** were a system I established as an update to the Research system, creating items that acted as single-use, respawnable elements. This was not only to add interest to the world,

but to create a risk-reward item for the player: the core object that spawned the Harvestables had a large Research Item with a lot of value beneath it, but to dig up that object (to get at the Research item) destroyed it.

For all of these items, I made sure to document their locations and details: what kind of Harvestables appeared on what planet, on what biome or level, how much its items could be worth, how long it took them to respawn, etc.; the Scrap values of each of the broken Discoveries and where they could be found (and the odds of finding them); and the species and subtypes of each enemy flora. All these systems overlapped, supporting (ideally) one another, and I wanted to make very sure that I not only had a good handle on such core elements of the game, but that all my work was well documented, so anyone brought in to support or update that work wasn't instantly struck into a coma at all the scattered information.







EXAMPLE #3 – *Destiny*, Public Events

When I joined the Public Events team – which was a great deal smaller than other sub-teams on the project, and more meant to be a bit of flavor rather than a focus – it was just before the big game reveal at E3 2013. After the gameplay reveal, the huge positive response to that part of the demo brought a lot of attention to that small team, and a lot was expected… which was especially difficult being that the restrictions for what the game could actually handle (even down to how many events per area and what type could be made) were all still up in the air. Thankfully, the team was up to it, and I did my best to bring all of the experience that I'd cultivated working on the first season of Guild Wars 2's Living World to play in this situation.

In addition to developing event types – such as the WARSAT or Fallen Extraction Crew – I aided in setting basic determinations for event location and difficulty. This was a team who had largely not worked in multiplayer settings before, and being able to approach an activity at any time from any angle made adapting placement in certain zones a unique challenge. We had to not only be mindful of any obstructions within the radius of the event itself, but how big its notification radius was, the line of sight on any key point(s), and where enemy spawns could happen without feeling either like they leapt right on top of players or showed up a mile away.

One of the more unique challenges I tackled in events was my coordination with the Writing team, who had until that point only worked on single-player and/or linear narrative games. I worked with them to establish not only stages of notification for events and what needed to be addressed (ex. 75% completed, boss nearly at end point, etc.) – but how those were different for a player who just entered an active event versus one who had been there from the start. This required a kind of rewiring of thinking that not only reflected who they needed to address, but also how long and pointed that address could be, forcing the shortening and punching up of lines that could be a luxuriant 6 seconds to focused and snappy alerts half that length or less.